

Docket No. 245902US2/im



IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

IN RE APPLICATION OF: Ayako KOBAYASHI

GAU: 2127

SERIAL NO: 10/723,603

EXAMINER:

FILED: November 26, 2003

FOR: IMAGE FORMING APPARATUS THAT CHECKS HARDWARE RESOURCES BEFORE
ACTIVATING HARDWARE-RELATED PROGRAMS

REQUEST FOR PRIORITY

COMMISSIONER FOR PATENTS
ALEXANDRIA, VIRGINIA 22313

SIR:

- ☐ Full benefit of the filing date of U.S. Application Serial Number , filed , is claimed pursuant to the provisions of 35 U.S.C. §120.
- ☐ Full benefit of the filing date(s) of U.S. Provisional Application(s) is claimed pursuant to the provisions of 35 U.S.C. §119(e): Application No. Date Filed

- ☒ Applicants claim any right to priority from any earlier filed applications to which they may be entitled pursuant to the provisions of 35 U.S.C. §119, as noted below.

In the matter of the above-identified application for patent, notice is hereby given that the applicants claim as priority:

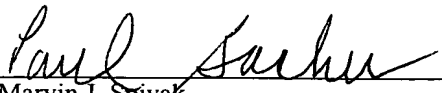
<u>COUNTRY</u>	<u>APPLICATION NUMBER</u>	<u>MONTH/DAY/YEAR</u>
JAPAN	2002-342826	November 26, 2002
JAPAN	2003-393414	November 25, 2003
JAPAN	2003-393415	November 25, 2003
JAPAN	2003-393416	November 25, 2003

Certified copies of the corresponding Convention Application(s)

- ☒ are submitted herewith
- ☐ will be submitted prior to payment of the Final Fee
- ☐ were filed in prior application Serial No. filed
- ☐ were submitted to the International Bureau in PCT Application Number
Receipt of the certified copies by the International Bureau in a timely manner under PCT Rule 17.1(a) has been acknowledged as evidenced by the attached PCT/IB/304.
- ☐ (A) Application Serial No.(s) were filed in prior application Serial No. filed ; and
- ☐ (B) Application Serial No.(s)
- ☐ are submitted herewith
- ☐ will be submitted prior to payment of the Final Fee

Respectfully Submitted,

OBLON, SPIVAK, McCLELLAND,
MAIER & NEUSTADT, P.C.


Marvin J. Spivak

Registration No. 24,913

Customer Number

22850

Tel. (703) 413-3000
Fax. (703) 413-2220
(OSMMN 05/03)

Paul Sacher
Registration No. 43,418

日 本 国 特 許 庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出 願 年 月 日 2 0 0 2 年 1 1 月 2 6 日
Date of Application:

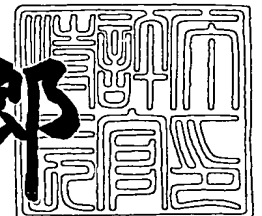
出 願 番 号 特 願 2 0 0 2 - 3 4 2 8 2 6
Application Number:
[ST: 10/C]: [J P 2 0 0 2 - 3 4 2 8 2 6]

出 願 人 株 式 会 社 リ コ ー
Applicant(s):

2 0 0 3 年 7 月 1 0 日

特許庁長官
Commissioner,
Japan Patent Office

太田信一郎



【書類名】 特許願

【整理番号】 0207551

【提出日】 平成14年11月26日

【あて先】 特許庁長官 太田 信一郎 殿

【国際特許分類】 G03G 21/00
G06F 17/60

【発明の名称】 画像形成装置およびプログラム起動方法

【請求項の数】 20

【発明者】

【住所又は居所】 東京都大田区中馬込 1 丁目 3 番 6 号 株式会社リコー内

【氏名】 小林 綾子

【特許出願人】

【識別番号】 000006747

【氏名又は名称】 株式会社リコー

【代理人】

【識別番号】 100070150

【弁理士】

【氏名又は名称】 伊東 忠彦

【手数料の表示】

【予納台帳番号】 002989

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 画像形成装置およびプログラム起動方法

【特許請求の範囲】

【請求項 1】 画像形成処理で使用されるハードウェア資源と、画像形成に係る処理を行うプログラムとを有する画像形成装置であって、

前記ハードウェア資源に関するチェックを行うチェック手段と、

前記チェック手段と前記プログラムとの関連を設定する設定手段と、

前記チェック手段によるチェックの結果に応じて前記チェック手段に関連する前記プログラムを起動する起動手段と

を有することを特徴とする画像形成装置。

【請求項 2】 前記設定手段は、前記チェック手段と前記プログラムとを 1 対 1 に関連付けて設定していることを特徴とする請求項 1 記載の画像形成装置。

【請求項 3】 前記設定手段は、前記チェック手段と前記プログラムとを 1 対 n に関連付けて設定していることを特徴とする請求項 1 記載の画像形成装置。

【請求項 4】 前記設定手段は、前記チェック手段と前記プログラムとを n 対 1 に関連付けて設定していることを特徴とする請求項 1 記載の画像形成装置。

【請求項 5】 前記チェック手段によるチェックの結果を格納しておく格納手段を更に有し、

前記チェック手段は、前記ハードウェア資源に関するチェックを行う前に前記格納手段に格納されているチェックの結果を確認し、これから行うチェックの結果が格納済みであれば前記チェックの結果を利用することを特徴とする請求項 1 記載の画像形成装置。

【請求項 6】 前記起動手段は、前記設定手段に設定されている内容に応じて前記チェック手段を起動することを特徴とする請求項 2 乃至 5 何れか一記載の画像形成装置。

【請求項 7】 前記起動手段は、前記プログラムを起動した後に、前記チェック手段を終了させることを特徴とする請求項 6 記載の画像形成装置。

【請求項 8】 前記チェック手段は、前記ハードウェア資源の有無をチェックし、前記ハードウェア資源が有ると判定したときに正常値をチェックの結果と

して出力し、前記ハードウェア資源が無いと判定したときに異常値をチェックの結果として出力することを特徴とする請求項 2 乃至 5 何れか一項記載の画像形成装置。

【請求項 9】 前記チェック手段は、前記ハードウェア資源に対応したデバイスドライバをオープンし、オープンが成功したとき又は既にオープンされていたときに前記ハードウェア資源が有ると判定し、それ以外の人に前記ハードウェア資源がないと判定することを特徴とする請求項 8 記載の画像形成装置。

【請求項 10】 前記チェック手段は、前記ハードウェア資源の性能をチェックし、前記ハードウェア資源が所定の性能を満たしていると判定したときに正常値をチェックの結果として出力し、前記ハードウェア資源が所定の性能を満たしていないと判定したときに異常値をチェックの結果として出力することを特徴とする請求項 2 乃至 5 何れか一項記載の画像形成装置。

【請求項 11】 前記起動手段は、前記チェックの結果が正常値であるときに前記チェック手段に関連付けて設定されている前記プログラムを起動し、前記チェックの結果が異常値であるときに前記チェック手段に関連付けて設定されている前記プログラムを起動しないことを特徴とする請求項 8 又は 10 記載の画像形成装置。

【請求項 12】 前記設定手段は、前記チェック手段と前記プログラムの存在するディレクトリ又は上位のディレクトリとを関連付けて設定しており、

前記起動手段は、前記チェックの結果が正常値であるときに前記チェック手段に関連付けて設定されている前記プログラムのディレクトリをマウントし、前記チェックの結果が異常値であるときに前記チェック手段に関連付けて設定されている前記プログラムのディレクトリをマウントしないことを特徴とする請求項 11 記載の画像形成装置。

【請求項 13】 前記格納手段は、前記チェック手段が直接アクセス可能なメモリ上の領域に作成されることを特徴とする請求項 5 記載の画像形成装置。

【請求項 14】 前記起動手段は、電源投入後に起動されたオペレーティングシステムにより起動されることを特徴とする請求項 1 記載の画像形成装置。

【請求項 15】 前記プログラムは、画像形成処理を行うアプリケーション

のプログラムと、画像形成処理で利用されるハードウェア資源の管理を行うコントロールサービスのプログラムと、オペレーティングシステムのプログラムとを有することを特徴とする請求項 1 記載の画像形成装置。

【請求項 16】 画像形成処理で利用されるハードウェア資源と、画像形成に係る処理を行うプログラムとを有する画像形成装置のプログラム起動方法であって、

起動手段が、前記プログラムと前記ハードウェア資源に関するチェックを行うチェック手段との関連を設定した設定手段を解析する解析段階と、

前記起動手段が、前記解析の結果に応じて前記チェック手段を起動するチェック手段起動段階と、

前記起動手段が、前記チェック手段によるチェックの結果に応じて前記チェック手段に関連する前記プログラムを起動するプログラム起動段階とを有することを特徴とするプログラム起動方法。

【請求項 17】 前記設定手段は、前記プログラムと前記チェック手段とを 1 対 1 に関連付けて設定していることを特徴とする請求項 16 記載のプログラム起動方法。

【請求項 18】 前記設定手段は、前記プログラムと前記チェック手段とを 1 対 n に関連付けて設定していることを特徴とする請求項 16 記載のプログラム起動方法。

【請求項 19】 前記設定手段は、前記プログラムと前記チェック手段とを n 対 1 に関連付けて設定していることを特徴とする請求項 16 記載のプログラム起動方法。

【請求項 20】 前記プログラム起動段階は、前記チェック手段が、チェックの結果を格納手段に格納するチェック結果格納段階と、

前記チェック手段が、前記ハードウェア資源に関するチェックを行う前に前記格納手段に格納されているチェックの結果を確認して、これから行うチェックの結果が前記格納手段に格納済みであれば前記チェックの結果を利用し、これから行うチェックの結果が前記格納手段に格納されていなければ前記ハードウェア資源に関するチェックを行うチェック段階と

を有することを特徴とする請求項 16 記載のプログラム起動方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は、画像形成装置およびプログラム起動方法に係り、特に所定の設定ファイルに応じてプログラムを起動する画像形成装置およびプログラム起動方法に関する。

【0002】

【従来の技術】

近年、プリンタ、コピー、ファクシミリおよびスキャナなどの各装置の機能を 1 つの筐体内に収納した画像形成装置（以下、融合機という）が知られるようになった。この融合機は、1 つの筐体内に表示部、印刷部および撮像部などを設けると共に、プリンタ、コピー、ファクシミリおよびスキャナにそれぞれ対応する 4 種類のソフトウェアを設け、そのソフトウェアを切り替えることより、プリンタ、コピー、ファクシミリおよびスキャナとして動作させるものである。

【0003】

このような融合機は、電源投入後に、BIOS (Basic Input/Output System) およびブートローダ (Boot Loader) が起動する。ブートローダは、カーネル (Kernel) およびルートファイルシステムを RAM (Random Access Memory) 上に展開してカーネルを起動する。そして、カーネルはルートファイルシステムをマウントする。ここでマウントとは、ファイルシステムや周辺機器などをアクセス可能な状態に起動することをいう。

【0004】

カーネルの起動後、アプリケーション（以下、アプリという）や各種サービスを起動するアプリ／サービス層起動プログラムが起動される。アプリ／サービス層起動プログラムは融合機で最初に起動されるプロセスであり、所定の設定ファイルに従ってファイルシステムをマウントし、融合機の動作に必要なサービス層およびアプリ層のプロセスを所定の設定ファイルに従って起動している。

【0005】

従来の融合機では、起動されたサービス層およびアプリ層のプロセスが、各プロセスの処理の中で表示部、印刷部および撮像部などのハードウェア資源のチェックを行っていた。

【0006】

【発明が解決しようとする課題】

しかしながら、従来の融合機ではハードウェア資源を複数のプロセスで共通に利用しており、各プロセスの処理の中でハードウェア資源のチェックを行うと各プロセスに重複した部分が多くなるという問題があった。

【0007】

また、従来の融合機では各プロセスの処理の中でハードウェア資源のチェックを行うため、各プロセスを起動しなければハードウェア資源の有無、性能などのチェックを行うことができなかった。したがって、従来の融合機ではハードウェア資源が存在しない、ハードウェア資源の性能が低い等の理由で使用しないプロセスであっても、ハードウェア資源のチェックを行うために無駄に起動しなければならないという問題があった。

【0008】

本発明は、上記の点に鑑みなされたもので、各プログラムの重複部分を削減することができ、ハードウェア資源に関連するプログラムを効率良く起動することが可能な画像形成装置およびプログラム起動方法を提供することを目的とする。

【0009】

【課題を解決するための手段】

そこで、上記課題を解決するため、本発明は、画像形成処理で使用されるハードウェア資源と、画像形成に係る処理を行うプログラムとを有する画像形成装置であって、前記ハードウェア資源に関するチェックを行うチェック手段と、前記チェック手段と前記プログラムとの関連を設定する設定手段と、前記チェック手段によるチェックの結果に応じて前記チェック手段に関連する前記プログラムを起動する起動手段とを有することを特徴とする。

【0010】

前記設定手段は、前記チェック手段と前記プログラムとを1対1に関連付けて

設定するようにしてもよい。

【0011】

前記設定手段は、前記チェック手段と前記プログラムとを1対nに関連付けて設定するようにしてもよい。

【0012】

前記設定手段は、前記チェック手段と前記プログラムとをn対1に関連付けて設定するようにしてもよい。

【0013】

前記チェック手段によるチェックの結果を格納しておく格納手段を更に有し、前記チェック手段は、前記ハードウェア資源に関するチェックを行う前に前記格納手段に格納されているチェックの結果を確認し、これから行うチェックの結果が格納済みであれば前記チェックの結果を利用するようにしてもよい。

【0014】

前記起動手段は、前記設定手段に設定されている内容に応じて前記チェック手段を起動するようにしてもよい。

【0015】

前記起動手段は、前記プログラムを起動した後に、前記チェック手段を終了させるようにしてもよい。

【0016】

前記チェック手段は、前記ハードウェア資源の有無をチェックし、前記ハードウェア資源が有ると判定したときに正常値をチェックの結果として出力し、前記ハードウェア資源が無いと判定したときに異常値をチェックの結果として出力するようにしてもよい。

【0017】

前記チェック手段は、前記ハードウェア資源に対応したデバイスドライバをオープンし、オープンが成功したとき又は既にオープンされていたときに前記ハードウェア資源が有ると判定し、それ以外のときに前記ハードウェア資源がないと判定するようにしてもよい。

【0018】

前記チェック手段は、前記ハードウェア資源の性能をチェックし、前記ハードウェア資源が所定の性能を満たしていると判定したときに正常値をチェックの結果として出力し、前記ハードウェア資源が所定の性能を満たしていないと判定したときに異常値をチェックの結果として出力するようにしてもよい。

【0019】

前記起動手段は、前記チェックの結果が正常値であるときに前記チェック手段に関連付けて設定されている前記プログラムを起動し、前記チェックの結果が異常値であるときに前記チェック手段に関連付けて設定されている前記プログラムを起動しないようにしてもよい。

【0020】

前記設定手段は、前記チェック手段と前記プログラムの存在するディレクトリ又は上位のディレクトリとを関連付けて設定しており、前記起動手段は、前記チェックの結果が正常値であるときに前記チェック手段に関連付けて設定されている前記プログラムのディレクトリをマウントし、前記チェックの結果が異常値であるときに前記チェック手段に関連付けて設定されている前記プログラムのディレクトリをマウントしないようにしてもよい。

【0021】

前記格納手段は、前記チェック手段が直接アクセス可能なメモリ上の領域に作成されるようにしてもよい。

【0022】

前記起動手段は、電源投入後に起動されたオペレーティングシステムにより起動されるようにしてもよい。

【0023】

前記プログラムは、画像形成処理を行うアプリケーションのプログラムと、画像形成処理で利用されるハードウェア資源の管理を行うコントロールサービスのプログラムと、オペレーティングシステムのプログラムとを有するようにしてもよい。

【0024】

また、上記課題を解決するため、本発明は、画像形成処理で使用されるハード

ウェア資源と、画像形成に係る処理を行うプログラムとを有する画像形成装置のプログラム起動方法であって、起動手段が、前記プログラムと前記ハードウェア資源に関するチェックを行うチェック手段との関連を設定した設定手段を解析する解析段階と、前記起動手段が、前記解析の結果に応じて前記チェック手段を起動するチェック手段起動段階と、前記起動手段が、前記チェック手段によるチェックの結果に応じて前記チェック手段に関連する前記プログラムを起動するプログラム起動段階とを有することを特徴とする。

【0025】

前記設定手段は、前記プログラムと前記チェック手段とを1対1に関連付けて設定するようにしてもよい。

【0026】

前記設定手段は、前記プログラムと前記チェック手段とを1対nに関連付けて設定するようにしてもよい。

【0027】

前記設定手段は、前記プログラムと前記チェック手段とをn対1に関連付けて設定するようにしてもよい。

【0028】

前記プログラム起動段階は、前記チェック手段が、チェックの結果を格納手段に格納するチェック結果格納段階と、前記チェック手段が、前記ハードウェア資源に関するチェックを行う前に前記格納手段に格納されているチェックの結果を確認して、これから行うチェックの結果が前記格納手段に格納済みであれば前記チェックの結果を利用し、これから行うチェックの結果が前記格納手段に格納されていなければ前記ハードウェア資源に関するチェックを行うチェック段階とを有するようにしてもよい。

【0029】

本発明によれば、ハードウェア資源に関するチェックをチェック手段で行うようにしたため、各プログラムの処理の中でハードウェア資源に関するチェックを行う必要が無くなり、各プログラムの重複部分を削減できる。

【0030】

また、本発明によれば、ハードウェア資源に関するチェックの結果に応じてプログラムを起動するため、使用しないプログラムを起動する必要がなくなり、プログラムを効率良く起動できる。

【0031】

【発明の実施の形態】

次に、本発明の実施の形態について図面に基づいて説明する。

【0032】

図1は、本発明による融合機のソフトウェア構成について説明するための一実施例の構成図である。融合機1は、ソフトウェア群2と、融合機起動部3と、ハードウェア資源4とを含むように構成される。

【0033】

ハードウェア資源4は、白黒レーザプリンタ（B&W LP）11と、カラーレーザプリンタ（Color LP）12と、スキャナやファクシミリなどのその他のハードウェアリソース13とを含む。また、ソフトウェア群2は、UNIX（登録商標）などのオペレーティングシステム（以下、OSという）上に起動されているアプリケーション層5とプラットフォーム6とを含む。

【0034】

アプリケーション層5は、プリンタ、コピー、ファックスおよびスキャナなどの画像形成にかかるユーザサービスにそれぞれ固有の処理を行うプログラムを含む。図1のアプリケーション層5は、プリンタアプリ21と、コピーアプリ22と、ファックスアプリ23と、スキャナアプリ24と、ネットファイルアプリ25とを含む。なお、ネットファイルアプリ25はネットワークファイル用アプリケーションであり、融合機1にネットワークを介して接続されるネットワーク機器とのデータ通信を管理するものである。

【0035】

また、プラットフォーム6は、アプリケーション層5からの処理要求を解釈してハードウェア資源4の獲得要求を発生するコントロールサービス層9と、1つ以上のハードウェア資源4の管理を行ってコントロールサービス層9からの獲得要求を調停するシステムリソースマネージャ（以下、SRMという）39と、S

RM39からの獲得要求に応じてハードウェア資源4の管理を行うハンドラ層10とを含む。

【0036】

図1のコントロールサービス層9は、ネットワークコントロールサービス（以下、NCSという）31，デリバリーコントロールサービス（以下、DCSという）32，オペレーションパネルコントロールサービス（以下、OCSという）33，ファックスコントロールサービス（以下、FCSという）34，エンジンコントロールサービス（以下、ECSという）35，メモリコントロールサービス（以下、MCSという）36，ユーザインフォメーションコントロールサービス（以下、UCSという）37，システムコントロールサービス（以下、SCSという）38など、一つ以上のサービスモジュールを含む。

【0037】

なお、プラットフォーム6は予め定義されている関数により、アプリケーション層5からの処理要求を受信するAPI53を有するように構成されている。OSは、アプリケーション層5およびプラットフォーム6の各ソフトウェアをプロセスとして並列実行する。

【0038】

NCS31のプロセスは、ネットワーク側から各プロトコルによって受信したデータを各アプリケーションに振り分けたり、各アプリケーションからのデータをネットワーク側に送信する際の仲介を行う。例えばNCS31は、融合機にネットワークを介して接続されるネットワーク機器とのデータ通信を制御する。

【0039】

DCS32のプロセスは、融合機に蓄積されている文書データの配送などの制御を行う。OCS33のプロセスは、オペレータとの間で情報伝達手段となるオペレーションパネルの制御を行う。FCS34のプロセスは、アプリケーション層5からPSTNまたはISDN網を利用したファックス送受信，バックアップ用のメモリで管理されている各種ファックスデータの登録／引用，ファックス読み取り，ファックス受信印刷などを行うためのAPIを提供する。

【0040】

ECS 35のプロセスは、白黒レーザプリンタ 11, カラーレーザプリンタ 12, ハードウェアリソース 13などのエンジン部の制御を行う。MCS 36のプロセスは、メモリの取得および解放, HDDの利用, 画像データの圧縮および伸張などのメモリ制御を行う。UCS 37のプロセスは、ユーザ情報の管理を行うものである。

【0041】

SCS 38のプロセスは、アプリケーション管理, 操作部制御, システム画面表示, LED表示, ハードウェア資源管理, 割り込みアプリケーション制御などの処理を行う。

【0042】

SRM 39のプロセスは、SCS 38と共にシステムの制御およびハードウェア資源 4 の管理を行うものである。例えばSRM 39のプロセスは、白黒レーザプリンタ 11やカラーレーザプリンタ 12などのハードウェア資源 4 を利用する上位層からの獲得要求に従って調停を行い、実行制御する。

【0043】

具体的に、SRM 39のプロセスは獲得要求されたハードウェア資源 4 が利用可能であるか（他の獲得要求により利用されていないかどうか）を判定し、利用可能であれば獲得要求されたハードウェア資源 4 が利用可能である旨を上位層に通知する。また、SRM 39のプロセスは上位層からの獲得要求に対してハードウェア資源 4 を利用するためのスケジューリングを行い、要求内容（例えば、プリンタエンジンによる紙搬送と作像動作, メモリ確保, ファイル生成など）を直接実施している。

【0044】

また、ハンドラ層 10は後述するファックスコントロールユニット（以下、FCUという）の管理を行うファックスコントロールユニットハンドラ（以下、FCUHという）40と、プロセスに対するメモリの割り振り及びプロセスに割り振ったメモリの管理を行うイメージメモリハンドラ（以下、IMHという）41とを含む。

【0045】

SRM39およびFCUH40は、予め定義されている関数によりハードウェア資源4に対する処理要求を送信するエンジンI/F54を利用して、ハードウェア資源4に対する処理要求を行う。

【0046】

図1の構成により、融合機1は各アプリケーションで共通的に必要な処理をプラットフォーム6で一元的に処理することができる。次に、融合機1のハードウェア構成について説明する。

【0047】

図2は、本発明による融合機のハードウェア構成について説明するための一実施例の構成図である。融合機1は、コントローラ60と、オペレーションパネル70と、FCU80と、USBデバイス90と、IEEE1394デバイス100と、エンジン部110とを含む。

【0048】

また、コントローラ60は、CPU61と、システムメモリ(MEM-P)62と、ノースブリッジ(以下、NBという)63と、サウスブリッジ(以下、SBという)64と、ASIC66と、ローカルメモリ(MEM-C)67と、HDD68およびネットワークインターフェースコントローラ69とを含む。

【0049】

オペレーションパネル70は、コントローラ60のASIC66に接続されている。また、FCU80、USBデバイス90、IEEE1394デバイス100およびエンジン部110は、コントローラ60のASIC66にPCIバスで接続されている。

【0050】

コントローラ60は、ASIC66にローカルメモリ67、HDD68およびネットワークインターフェースコントローラ69などが接続されると共に、CPU61とASIC66とがCPUチップセットのNB63を介して接続されている。このように、NB63を介してCPU61とASIC66とを接続することにより、CPU61のインターフェースが公開されていない場合に対応する。

【0051】

なお、ASIC 66 と NB 63 とは PCI バスを介して接続されているのではなく、AGP (Accelerated Graphics Port) 65 を介して接続されている。このように、図 1 のアプリケーション層 5 やプラットフォーム 6 を形成する一つ以上のプロセスを実行制御するため、ASIC 66 と NB 63 とを低速の PCI バスでなく AGP 65 を介して接続し、パフォーマンスの低下を防いでいる。

【0052】

CPU 61 は、融合機 1 の全体制御を行うものである。CPU 61 は、NCS 31, DCS 32, OCS 33, FCS 34, ECS 35, MCS 36, UCS 37, SCS 38, SRM 39, FCUH 40 および IMH 41 を OS 上にそれぞれプロセスとして起動して実行させると共に、アプリケーション層 5 を形成するプリンタアプリ 21, コピーアプリ 22, ファックスアプリ 23, スキャナアプリ 24, ネットファイルアプリ 25 を起動して実行させる。

【0053】

NB 63 は、CPU 61, システムメモリ 62, SB 64 および ASIC 66 を接続するためのブリッジである。システムメモリ 62 は、融合機 1 の描画用メモリなどとして用いるメモリである。SB 64 は、NB 63 と ROM, PCI バス, 周辺デバイスとを接続するためのブリッジである。また、ローカルメモリ 67 はコピー用画像バッファ, 符号化用バッファなどとして用いるメモリである。

【0054】

ASIC 66 は、画像処理用のハードウェア要素を有する画像処理用途向けの IC である。HDD 68 は、画像データの蓄積, 文書データの蓄積, プログラムの蓄積, フォントデータの蓄積, フォームの蓄積などを行うためのストレージである。ネットワークインターフェースコントローラ 69 は、ネットワークを介して接続されているネットワーク機器と MAC アドレスなどを用いて通信する。また、オペレーションパネル 70 は、オペレータからの入力操作を受け付けると共に、オペレータに向けた表示を行う操作部である。

【0055】

図 3 は、融合機起動部の一例の構成図である。図 1 の融合機起動部 3 は、融合機 1 の電源投入時に最初に実行され、アプリケーション層 5 やプラットフォーム

6を起動するものである。融合機起動部50は、ROMモニタ51と、プログラム起動部52とを含む。

【0056】

ここで、融合機起動部3の処理について図4のフローチャートを参照しつつ説明する。図4は、融合機起動部の処理の一例のフローチャートである。ステップS1では、融合機1の電源ONにより、BIOSおよびブートローダとしてのROMモニタ51が実行される。ROMモニタ51は、ハードウェアの初期化、コントローラ60の診断、ソフトウェアの初期化などを行う。

【0057】

ステップS1に続いてステップS2に進み、ROMモニタ51は、OSおよびルートファイルシステムをシステムメモリ62上に展開してOSを起動する。そして、OSはルートファイルシステムをマウントする。

【0058】

ステップS2に続いてステップS3に進み、OSは起動時に接続されたデバイスのデバイス情報（例えば、CPU61のクロック周波数、システムメモリ62およびローカルメモリ57のメモリサイズ、コントローラ60のボードタイプなど）を取得する。

【0059】

ステップS3に続いてステップS4に進み、OSはアプリ／サービス起動プログラムとしてのプログラム起動部52を起動する。プログラム起動部52はシステムメモリ62およびローカルメモリ67上にメモリ領域を確保する。プログラム起動部52は、融合機1で最初に起動されるプロセスである。

【0060】

ステップS4に続いてステップS5に進み、プログラム起動部52は設定ファイルに従ってファイルシステムをマウントする。また、プログラム起動部52は設定ファイルに従って、ハードウェア資源に関するチェック手段としてのチェックプログラムを起動する。プログラム起動部52は、チェックプログラムが正常に終了したか否かで、設定ファイルに記述されたアプリケーション層5およびプラットフォーム6のプログラム（以下、本体プログラムという）を起動するか否

かを判定する。

【0061】

本体プログラムを起動すると判定した場合、プログラム起動部52は、その本体プログラムを設定ファイルに従ってROMなどから読み出し、読み出した本体プログラムをシステムメモリ62、ローカルメモリ67上に確保したメモリ領域に展開して、アプリケーション層5およびプラットフォーム6のプロセスを起動する。

(第1実施例)

更に、プログラム起動部52が行うステップS5の処理について詳細に説明していく。図5は、プログラム起動部の処理の一例のフローチャートである。

【0062】

ステップS10では、プログラム起動部52が設定ファイルを解析する。ステップS10に続いてステップS11に進み、プログラム起動部52は設定ファイルに従ってファイルシステムをマウントする。

【0063】

ステップS11に続いてステップS12に進み、プログラム起動部52は設定ファイルに記述されたexecコマンドを1つ読み出し、読み出したexecコマンドに「-c」オプションがあるか否かを判定する。例えば図6のような設定ファイルの場合、プログラム起動部52は1行目のexecコマンドに「-c」オプションがあると判定する。

【0064】

execコマンドに「-c」オプションがあると判定すると（S12においてYES）、プログラム起動部52はステップS13に進み、execコマンドで指定されたチェックプログラムを起動する。例えば図6のような設定ファイルの場合、プログラム起動部52は1行目のexecコマンドで指定されたチェックプログラム「fcucheck」を起動する。

【0065】

起動されたチェックプログラムは、後述するように、ハードウェア資源に関するチェック（例えばハードウェア資源の有無、性能などのチェック）を行い、チ

チェックの結果をプログラム起動部 52 に通知する。

【0066】

ステップ S13 に続いてステップ S14 に進み、プログラム起動部 52 はチェックプログラムから通知されたチェックの結果に基づいて、チェックプログラムが正常終了したか異常終了したかを判定する。チェックプログラムが正常終了したと判定すると（S14 において YES）、プログラム起動部 52 はステップ S15 に進む。

【0067】

ステップ S15 では、プログラム起動部 52 が、exec コマンドで指定された本体プログラムを起動する。例えば図 6 のような設定ファイルの場合、プログラム起動部 52 は 1 行目の exec コマンドで指定された本体プログラム「/fax/bin/fax」を起動する。

【0068】

ステップ S15 に続いてステップ S16 に進み、プログラム起動部 52 は起動すべき本体プログラム、言い換えれば読み込んでいない exec コマンドが設定ファイルにまだ存在するか否かを判定する。

【0069】

まだ読み込んでいない exec コマンドが設定ファイルに存在すると判定すると（S16 において YES）、プログラム起動部 52 はステップ S12 に戻り、まだ読み込んでいない exec コマンドを設定ファイルから読み込んでステップ S12 以降の処理を行う。一方、まだ読み込んでいない exec コマンドが設定ファイルに存在しないと判定すると（S16 において NO）、プログラム起動部 52 は処理を終了する。

【0070】

なお、ステップ S12 において、exec コマンドに「-c」オプションがないと判定すると（S12 において NO）、プログラム起動部 52 はステップ S15 に進み、exec コマンドで指定された本体プログラムを起動する。

【0071】

即ち、exec コマンドに「-c」オプションがなければ、プログラム起動部

52は必ずexecコマンドで指定された本体プログラムを起動する。

【0072】

また、ステップS14において、チェックプログラムが正常終了していないと判定すると（S14においてNO）、プログラム起動部52はステップS12に戻り、まだ読み込んでいないexecコマンドを設定ファイルから読み込んでステップS12以降の処理を行う。即ち、チェックプログラムが異常終了すると、プログラム起動部52はexecコマンドで指定された本体プログラムを起動しない。

【0073】

図5のフローチャートの処理により、プログラム起動部52はチェックプログラムが正常終了したときにexecコマンドで指定されたプログラムを起動する一方、チェックプログラムが異常終了したときにexecコマンドで指定されたプログラムを起動しないため、画像形成装置1の動作を抑制できる。

【0074】

次に、図6の設定ファイルを用いてチェックプログラムの処理について説明していく。図6の設定ファイルでは、1行目のexecコマンドに「-c」オプションがあるので、チェックプログラム「fcucheck」が起動される。

【0075】

プログラム起動部52により起動されたチェックプログラム「fcucheck」は、例えば図7のような処理を行う。図7は、チェックプログラム「fcucheck」の処理の一例のフローチャートである。ステップS20では、チェックプログラムがFCU80のデバイスドライバをオープンする。

【0076】

ステップS20に続いてステップS21に進み、チェックプログラムはステップS20でのデバイスドライバのオープンが成功したか否かを判定する。デバイスドライバのオープンが成功したと判定すると（S21においてYES）、チェックプログラムはステップS23に進み、FCU80が融合機1に接続されているとみなして、正常終了を表す値「0」をプログラム起動部52に通知する。

【0077】

一方、デバイスドライバのオープンが失敗したと判定すると（S21においてNO）、チェックプログラムはステップS22に進み、FCU80が既にオープンされてビジー状態であるか否かを判定する。なお、FCU80がビジー状態であるか否かは、例えば「error」に「EBUSY」が入っているか否かで確認できる。

【0078】

FCU80が既にオープンされてビジー状態であると判定すると（S22においてYES）、チェックプログラムはステップS24に進み、FCU80が融合機1に接続されているとみなして、正常終了を表す値「0」をプログラム起動部52に通知する。一方、FCU80が既にオープンされてビジー状態であると判定しなかった場合（S22においてNO）、チェックプログラムはステップS25に進み、FCU80が融合機1に接続されていないとみなして、異常終了を表す値「1」をプログラム起動部52に通知する。

【0079】

図7のフローチャートの処理により、チェックプログラムはFCU80が融合機1に接続されている場合に正常終了をプログラム起動部52に通知し、FCU80が融合機1に接続されていない場合に異常終了をプログラム起動部52に通知できる。

【0080】

なお、プログラム起動部52は、チェックプログラムから正常終了を通知されたときにアプリ「fax」を起動し、チェックプログラムから異常終了を通知されたときにアプリ「fax」を起動しない。

【0081】

したがって、プログラム起動部52はチェックプログラムから通知される正常終了または異常終了を利用することで、FCU80が融合機1に接続されている場合にアプリ「fax」を起動し、接続されていない場合にアプリ「fax」を起動しないようにできる。即ち、プログラム起動部52はアプリ「fax」の起動をFCU80の接続状態によって抑制できる。

【0082】

なお、図7のフローチャートではデバイスの一例としてFCU80を用いて説明したが、如何なるデバイスであってもよい。つまり、図7のフローチャートによれば、オプションボードのようなデバイスの接続状態によってアプリの起動を抑制することができる。

【0083】

図6の設定ファイルでは、1行目のexecコマンドの処理が終了すると、2行目のexecコマンドの処理が行われる。図6の設定ファイルでは、2行目のexecコマンドに「-c」オプションがあるので、チェックプログラム「cpucheck1」が起動される。

【0084】

プログラム起動部52により起動されたチェックプログラム「cpucheck1」は、例えば図8のような処理を行う。図8は、チェックプログラム「cpucheck1」の処理の一例のフローチャートである。

【0085】

ステップS30では、チェックプログラムが、システムコール「getINFO(CPU)」を呼び、デバイス情報に含まれるCPU61のクロック周波数をOSから取得する。

【0086】

ステップS30に続いてステップS31に進み、チェックプログラムはステップS30で取得したCPU61のクロック周波数が500MHz以下であるか否かを判定する。

【0087】

CPU61のクロック周波数が500MHz以下であると判定すると(S31においてYES)、チェックプログラムはステップS32に進み、正常終了を表す値「0」をプログラム起動部52に通知する。一方、CPU61のクロック周波数が500MHz以下でないと判定すると(S31においてNO)、チェックプログラムはステップS33に進み、異常終了を表す値「1」をプログラム起動部52に通知する。

【0088】

図8のフローチャートの処理により、チェックプログラムはCPU61のクロック周波数が500MHz以下である場合に正常終了をプログラム起動部52に通知し、CPU61のクロック周波数が500MHz以下でない場合に異常終了をプログラム起動部52に通知できる。

【0089】

なお、プログラム起動部52は、チェックプログラムから正常終了を通知されたときにアプリ「setfont__bitmap」を起動し、チェックプログラムから異常終了を通知されたときにアプリ「setfont__bitmap」を起動しない。

【0090】

したがって、プログラム起動部52はチェックプログラムから通知される正常終了または異常終了を利用することで、プリンタが使用するデフォルトフォントをビットマップ形式に設定する。即ち、プログラム起動部52はCPU61のクロック周波数が500MHz以下であっても、一定時間内にプリントできるようなデフォルトフォントに変えることで、プリントの高速性を優先できる。

【0091】

図6の設定ファイルでは、2行目のexecコマンドの処理が終了すると、3行目のexecコマンドの処理が行われる。図6の設定ファイルでは、3行目のexecコマンドに「-c」オプションがあるので、チェックプログラム「cpucheck2」が起動される。

【0092】

プログラム起動部52により起動されたチェックプログラム「cpucheck2」は、例えば図9のような処理を行う。図9は、チェックプログラム「cpucheck2」の処理の一例のフローチャートである。

【0093】

ステップS40では、チェックプログラムが、システムコール「getINFO(CPU)」を呼び、デバイス情報に含まれるCPU61のクロック周波数をOSから取得する。

【0094】

ステップS40に続いてステップS41に進み、チェックプログラムはステップS40で取得したCPU61のクロック周波数が501MHz以上であるか否かを判定する。

【0095】

CPU61のクロック周波数が501MHz以上であると判定すると（S41においてYES）、チェックプログラムはステップS42に進み、正常終了を表す値「0」をプログラム起動部52に通知する。一方、CPU61のクロック周波数が501MHz以上でないと判定すると（S41においてNO）、チェックプログラムはステップS43に進み、異常終了を表す値「1」をプログラム起動部52に通知する。

【0096】

図9のフローチャートの処理により、チェックプログラムはCPU61のクロック周波数が501MHz以上である場合に正常終了をプログラム起動部52に通知し、CPU61のクロック周波数が501MHz以上でない場合に異常終了をプログラム起動部52に通知できる。

【0097】

なお、プログラム起動部52は、チェックプログラムから正常終了を通知されたときにアプリ「setfont__vector」を起動し、チェックプログラムから異常終了を通知されたときにアプリ「setfont__vector」を起動しない。

【0098】

したがって、プログラム起動部52はチェックプログラムから通知される正常終了または異常終了を利用することで、プリンタが使用するデフォルトフォントをベクトル画像に設定する。即ち、プログラム起動部52はCPU61のクロック周波数が501MHz以上であるときに、精細なデフォルトフォントに変えることで、プリントの高画質化が可能である。

【0099】

図8および図9のフローチャートの処理により、本発明の融合機1は一定の時間内にプリントできるように、CPU61の性能に応じて用いるデフォルトフォ

ントを変えることで、プリントの高速性を優先することができる。

【0100】

図6の設定ファイルでは、3行目のexecコマンドの処理が終了すると、4行目のexecコマンドの処理が行われる。図6の設定ファイルでは、4行目のexecコマンドに「-c」オプションがあるので、チェックプログラム「memcheck1」が起動される。

【0101】

プログラム起動部52により起動されたチェックプログラム「memcheck1」は、例えば図10のような処理を行う。図10は、チェックプログラム「memcheck1」の処理の一例のフローチャートである。

【0102】

ステップS50では、チェックプログラムが、システムコール「getINFO(mem)」を呼び、デバイス情報に含まれるシステムメモリ62及びローカルメモリ67のメモリサイズをOSから取得する。

【0103】

ステップS50に続いてステップS51に進み、チェックプログラムはステップS50で取得したメモリサイズが64MB以上128MB以下であるか否かを判定する。

【0104】

メモリサイズが64MB以上128MB以下であると判定すると(S51においてYES)、チェックプログラムはステップS52に進み、正常終了を表す値「0」をプログラム起動部52に通知する。一方、メモリサイズが64MB以上128MB以下でないと判定すると(S51においてNO)、チェックプログラムはステップS53に進み、異常終了を表す値「1」をプログラム起動部52に通知する。

【0105】

図10のフローチャートの処理により、チェックプログラムはメモリサイズが64MB以上128MB以下である場合に正常終了をプログラム起動部52に通知し、メモリサイズが64MB以上128MB以下でない場合に異常終了をプロ

グラム起動部52に通知できる。

【0106】

なお、プログラム起動部52はチェックプログラムから正常終了を通知されたときにhttpのデーモン（以下、httpdという）を5個起動し、チェックプログラムから異常終了を通知されたときにhttpdを起動しない。

【0107】

したがって、プログラム起動部52はチェックプログラムから通知される正常終了または異常終了を利用することで、システムメモリ62及びローカルメモリ67のメモリサイズが小さい場合であっても、起動するhttpdの数を少なくして、一つの要求に対する処理時間の増大を防ぐことができる。

【0108】

図6の設定ファイルでは、4行目のexecコマンドの処理が終了すると、5行目のexecコマンドの処理が行われる。図6の設定ファイルでは、5行目のexecコマンドに「-c」オプションがあるので、チェックプログラム「memcheck2」が起動される。

【0109】

プログラム起動部52により起動されたチェックプログラム「memcheck2」は、例えば図11のような処理を行う。図11は、チェックプログラム「memcheck2」の処理の一例のフローチャートである。

【0110】

ステップS60では、チェックプログラムが、システムコール「getINFO(mem)」を呼び、デバイス情報に含まれるシステムメモリ62及びローカルメモリ67のメモリサイズをOSから取得する。

【0111】

ステップS60に続いてステップS61に進み、チェックプログラムはステップS60で取得したメモリサイズが128MB以上であるか否かを判定する。メモリサイズが128MB以上であると判定すると（S61においてYES）、チェックプログラムはステップS62に進み、正常終了を表す値「0」をプログラム起動部52に通知する。一方、メモリサイズが128MB以上でないと判定す

ると (S 6 1 において NO)、チェックプログラムはステップ S 6 3 に進み、異常終了を表す値「1」をプログラム起動部 5 2 に通知する。

【0112】

図 1 1 のフローチャートの処理により、チェックプログラムはメモリサイズが 1 2 8 MB 以上である場合に正常終了をプログラム起動部 5 2 に通知し、メモリサイズが 1 2 8 MB 以上でない場合に異常終了をプログラム起動部 5 2 に通知できる。

【0113】

なお、プログラム起動部 5 2 はチェックプログラムから正常終了を通知されたときに h t t p d を 1 0 個起動し、チェックプログラムから異常終了を通知されたときに h t t p d を起動しない。

【0114】

したがって、プログラム起動部 5 2 はチェックプログラムから通知される正常終了または異常終了を利用することで、システムメモリ 6 2 及びローカルメモリ 6 7 のメモリサイズが大きい場合に、起動する h t t p d の数を多くして、クライアントからの要求にすぐ対応することができる。

【0115】

図 1 0 および図 1 1 のフローチャートの処理により、本発明の融合機 1 はシステムメモリ 6 2 及びローカルメモリ 6 7 のメモリサイズに応じて、起動する h t t p s の数を適切に変えることができる。

【0116】

図 6 の設定ファイルの場合、本体プログラムと、チェックプログラムと、プログラム起動部 5 2 と、OS と、ハードウェアとの関係は、例えば図 1 2 のようになる。図 1 2 は、本体プログラムと、チェックプログラムと、プログラム起動部 5 2 と、OS と、ハードウェアとの関係を表した一例の関係図である。

【0117】

プログラム起動部 5 2 はチェックプログラムを順番に起動し、チェックプログラムが正常終了したとき、そのチェックプログラムに対応する本体プログラムを起動する。図 1 2 では、各チェックプログラムの上に位置している本体プログラ

ムがチェックプログラムに対応する本体プログラムである。

(第2実施例)

図12の例では、チェックプログラムと本体プログラムとが1対1に対応しているが、図13のようにチェックプログラムと本体プログラムとが1対nに対応してもよい。

【0118】

図13は、チェックプログラムと本体プログラムとが1対nに対応する例について説明するための図である。図13の例では、チェックプログラムaと本体プログラムa、bとが対応している。プログラム起動部52はチェックプログラムaを起動し、チェックプログラムaが正常終了したとき、チェックプログラムaに対応している本体プログラムa、bを起動する。

【0119】

図13のようにチェックプログラムaと本体プログラムa、bとが対応している場合、設定ファイルは図14のように構成される。図14は、設定ファイルの他の一例の構成図である。なお、プログラム起動部52の処理は図5と同様であるので説明を省略する。

【0120】

また、同じチェックプログラムによるチェックの結果に基づき、ディレクトリ下の複数の本体プログラムを起動するような場合、チェックの結果に基づいてディレクトリのマウントを抑制すると効率的である。

【0121】

図15は、マウントを抑制する設定ファイルの一例の構成図である。図15のような設定ファイルの場合、プログラム起動部52は1行目のmountコマンドで指定されたチェックプログラム「memcheck3」を起動する。例えば図15のチェックプログラム「memcheck3」は、システムメモリ62及びローカルメモリ67のメモリサイズが64MB以上のときに正常終了するものとする。

【0122】

プログラム起動部52は、チェックプログラムが正常終了するとmountコ

マンドで指定された「web. romfs」をディレクトリ「/web」へマウントする一方、チェックプログラムが異常終了するとマウントしない。

【0123】

例えば図15の設定ファイルの記述の下に図6のような記述があれば、ディレクトリ「/web」以下のプログラム「/web/bin/httpd」なども起動されなくなる。したがって、プログラム起動部52はシステムメモリ62及びローカルメモリ67のメモリサイズに応じて無駄にメモリを消費することを避けることができる。

【0124】

なお、図15の設定ファイルによるマウント処理は、図5のステップS11で行われる処理である。また、図15の設定ファイルでは、メモリサイズに応じてディレクトリのマウントを抑制する例について表しているが、ハードウェア資源の有無やCPU性能などに応じてディレクトリのマウントを抑制するようにしてもよい。

(第3実施例)

図13の例では、チェックプログラムと本体プログラムとが1対nに対応しているが、図16のようにチェックプログラムと本体プログラムとがn対1に対応してもよい。

【0125】

図16は、チェックプログラムと本体プログラムとがn対1に対応する例について説明するための図である。図16の例では、チェックプログラムa, bと本体プログラムaとが対応している。プログラム起動部52はチェックプログラムa, bを起動し、チェックプログラムa, bが正常終了したとき、チェックプログラムa, bに対応している本体プログラムaを起動する。

【0126】

図16のようにチェックプログラムa, bと本体プログラムaとが対応している場合、設定ファイルは図17のように構成される。図17は、設定ファイルの他の一例の構成図である。

【0127】

図17の設定ファイルでは、1つのexecコマンドに「-c」オプションのある2つのチェックプログラム「チェックプログラムa」, 「チェックプログラムb」が設定されている。したがって、プログラム起動部52は図5のステップS13でチェックプログラムa, bを起動する。そして、プログラム起動部52はステップS14でチェックプログラムa, bが両方とも正常終了したか否かを判定し、両方とも正常終了したと判定したときにexecコマンドで指定された本体プログラムaを起動する。なお、プログラム起動部52のステップS13及びS14以外の処理は図5と同様であるので説明を省略する。

(第4実施例)

融合機1では、設定ファイルに応じて、同一のチェックプログラムが起動されることもある。したがって、以前に起動したチェックプログラムを再び起動する場合は、そのチェックプログラムのチェックの結果を利用する方が処理時間の短縮の観点から望ましい。そこで、本発明の融合機1は図18のような処理を行うことにより、以前に起動されたチェックプログラムのチェックの結果を利用できるようにする。

【0128】

図18及び図19は、プログラム起動部およびチェックプログラムが行う処理の一例のフローチャートである。なお、ステップS70～S73までの処理は図5のステップS10～S13の処理と同様であり、説明を省略する。

【0129】

ステップS74では、ステップS73で起動されたチェックプログラムが、所定のメモリ領域にチェックの結果が格納されているか否かでチェック済みであるか否かを判定する。例えばチェックの結果を格納する所定のメモリ領域には、OSを介することなく、チェックプログラムのプロセスが直接アクセス可能なメモリ上の領域を利用することができる。

【0130】

チェック済みであると判定すると(S74においてYES)、チェックプログラムは所定のメモリ領域からチェックの結果を読み出してプログラム起動部52に通知する。

【0131】

一方、チェック済みでないと判定すると（S74においてNO）、チェックプログラムはステップS76に進み、前述したようなハードウェア資源に関するチェックを行う。ステップS76に続いてステップS77に進み、チェックプログラムはチェックの結果を所定のメモリ領域に書き込むと共に、チェックの結果をプログラム起動部52に通知する。

【0132】

なお、ステップS78～S80の処理は図5のステップS14～S16の処理と同様であり、説明を省略する。図18及び図19のフローチャートにより、以前に起動されたチェックプログラムのチェックの結果を利用できる。

【0133】

さらに、融合機1は起動時に全てのチェックプログラムを起動し、全てのチェックプログラムによるチェックの結果を所定のメモリ領域に予め書き込んでおくことも考えられる。この場合、融合機1は起動直後にチェックプログラムによるチェックを行ったあと、所定のメモリ領域に書き込まれたチェックの結果を利用することができるので、処理時間の短縮が可能となる。

【0134】

本発明は、具体的に開示された実施例に限定されるものではなく、特許請求の範囲から逸脱することなく、種々の変形や変更が可能である。

【0135】**【発明の効果】**

上述の如く、本発明によれば、ハードウェア資源に関するチェックをチェック手段で行うようにしたため、各プログラムの処理の中でハードウェア資源に関するチェックを行う必要が無くなり、各プログラムの重複部分を削減できる。

【0136】

また、本発明によれば、ハードウェア資源に関するチェックの結果に応じてプログラムを起動するため、使用しないプログラムを起動する必要がなくなり、プログラムを効率良く起動できる。

【0137】

【図面の簡単な説明】

【図 1】

本発明による融合機のソフトウェア構成について説明するための一実施例の構成図である。

【図 2】

本発明による融合機のハードウェア構成について説明するための一実施例の構成図である。

【図 3】

融合機起動部の一例の構成図である。

【図 4】

融合機起動部の処理の一例のフローチャートである。

【図 5】

プログラム起動部の処理の一例のフローチャートである。

【図 6】

設定ファイルの一例の構成図である。

【図 7】

チェックプログラム「fcucHECK」の処理の一例のフローチャートである。

【図 8】

チェックプログラム「cpucHECK1」の処理の一例のフローチャートである。

【図 9】

チェックプログラム「cpucHECK2」の処理の一例のフローチャートである。

【図 10】

チェックプログラム「memCHECK1」の処理の一例のフローチャートである。

【図 11】

チェックプログラム「memCHECK2」の処理の一例のフローチャートで

ある。

【図 12】

本体プログラムと、チェックプログラムと、プログラム起動部 52 と、OS と、ハードウェアとの関係を表した一例の関係図である。

【図 13】

チェックプログラムと本体プログラムとが 1 対 n に対応する例について説明するための図である。

【図 14】

設定ファイルの他の一例の構成図である。

【図 15】

マウントを抑制する設定ファイルの一例の構成図である。

【図 16】

チェックプログラムと本体プログラムとが n 対 1 に対応する例について説明するための図である。

【図 17】

設定ファイルの他の一例の構成図である。

【図 18】

プログラム起動部およびチェックプログラムが行う処理の一例のフローチャートである (1/2)。

【図 19】

プログラム起動部およびチェックプログラムが行う処理の一例のフローチャートである (2/2)。

【符号の説明】

- 1 融合機
- 2 ソフトウェア群
- 3 融合機起動部
- 4 ハードウェア資源
- 5 アプリケーション層
- 6 プラットフォーム

- 9 コントロールサービス層
 - 10 ハンドラ層
 - 11 白黒レーザプリンタ (B&W LP)
 - 12 カラーレーザプリンタ (Color LP)
 - 13 ハードウェアリソース
 - 21 プリンタアプリ
 - 22 コピーアプリ
 - 23 ファックスアプリ
 - 24 スキャナアプリ
 - 25 ネットファイルアプリ
 - 31 ネットワークコントロールサービス (NCS)
 - 32 デリバリーコントロールサービス (DCS)
 - 33 オペレーションパネルコントロールサービス (OCS)
 - 34 ファックスコントロールサービス (FCS)
 - 35 エンジンコントロールサービス (ECS)
 - 36 メモリコントロールサービス (MCS)
 - 37 ユーザインフォメーションコントロールサービス (UCS)
 - 38 システムコントロールサービス (SCS)
 - 39 システムリソースマネージャ (SRM)
 - 40 ファックスコントロールユニットハンドラ (FCUH)
 - 41 イメージメモリハンドラ (IMH)
 - 51 ROMモニタ
 - 52 プログラム起動部
 - 53 アプリケーションプログラムインターフェース (API)
 - 54 エンジン I/F
- 60 コントローラ
 - 61 CPU
 - 62 システムメモリ
 - 63 ノースブリッジ (NB)

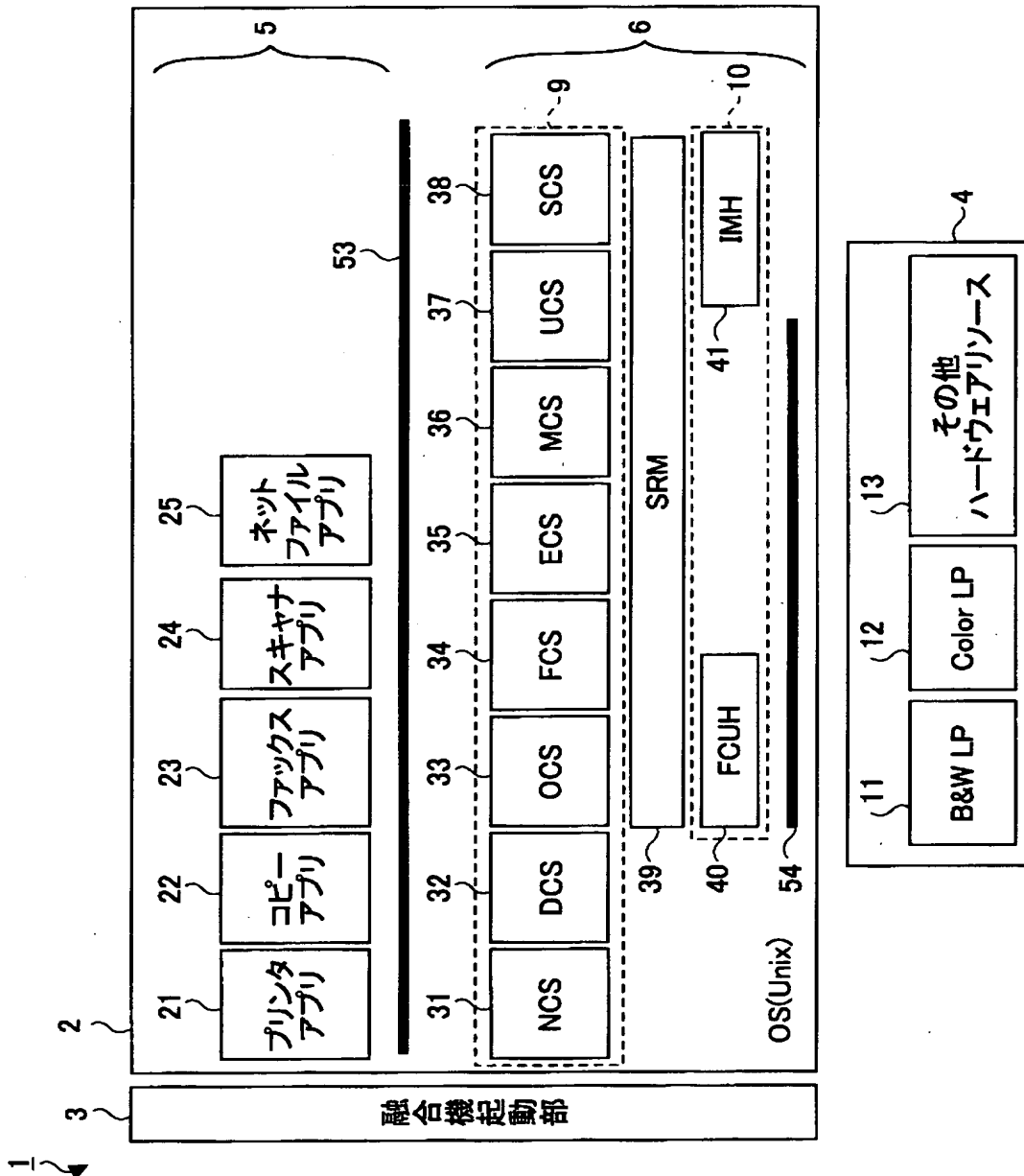
- 6 4 サウスブリッジ (S B)
- 6 5 A G P (Accelerated Graphics Port)
- 6 6 A S I C
- 6 7 ローカルメモリ
- 6 8 ハードディスク装置 (H D D)
- 6 9 ネットワークインターフェースコントローラ
- 7 0 オペレーションパネル
- 8 0 ファックスコントロールユニット (F C U)
- 9 0 U S B デバイス
- 1 0 0 I E E E 1 3 9 . 4 デバイス
- 1 1 0 エンジン部

【書類名】

図面

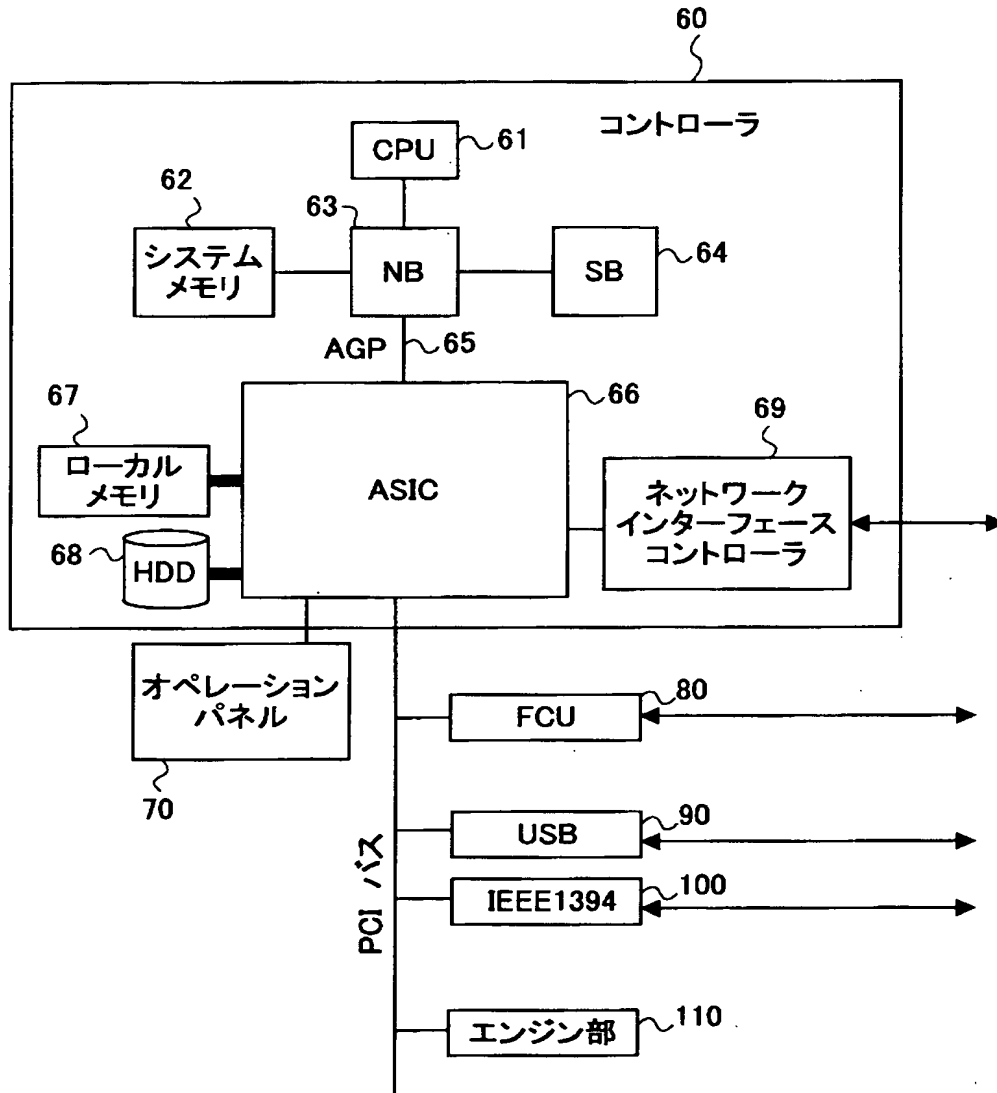
【図 1】

本発明による融合機のソフトウェア構成について説明するための一実施例の構成図



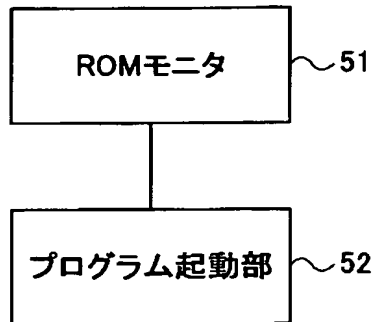
【図 2】

本発明による融合機のハードウェア構成について説明するための一実施例の構成図



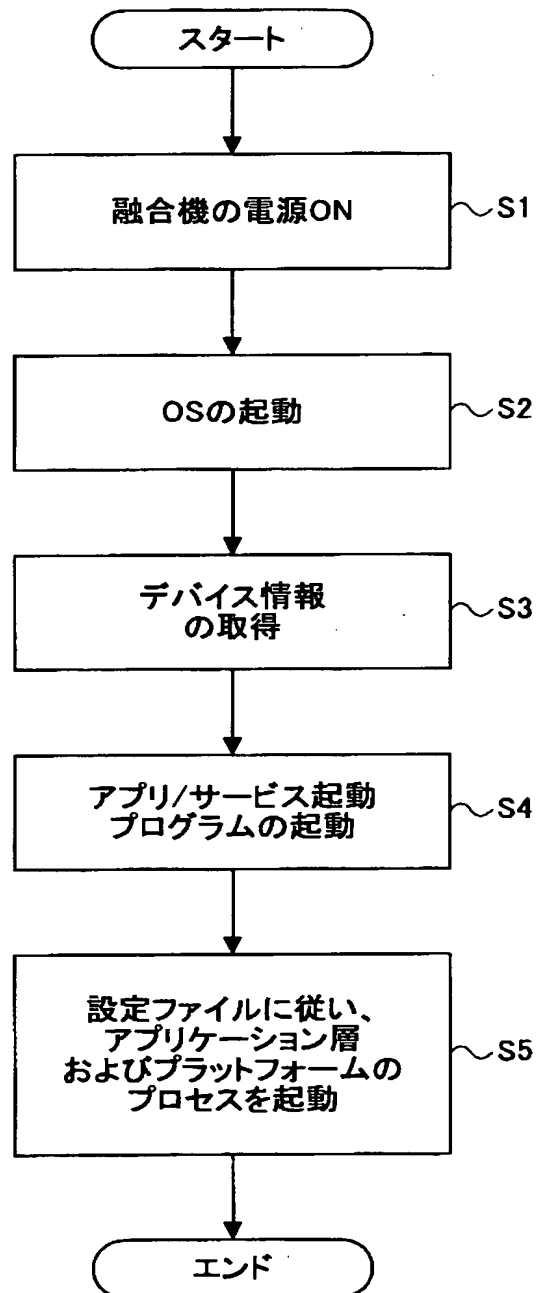
【図 3】

融合機起動部の一例の構成図

3

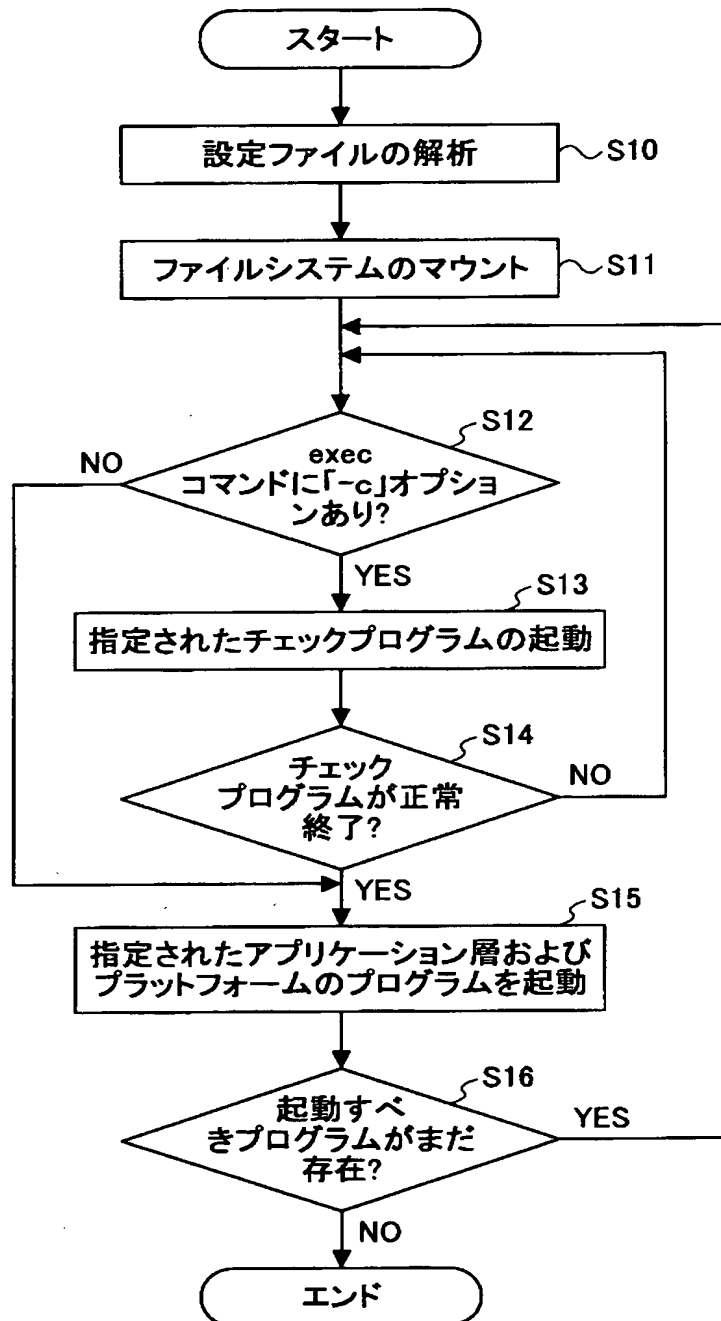
【図 4】

融合機起動部の処理の一例のフローチャート



【図 5】

プログラム起動部の処理の一例のフローチャート



【図 6】

設定ファイルの一例の構成図

```
exec -c fcucheck /fax/bin/fax
```

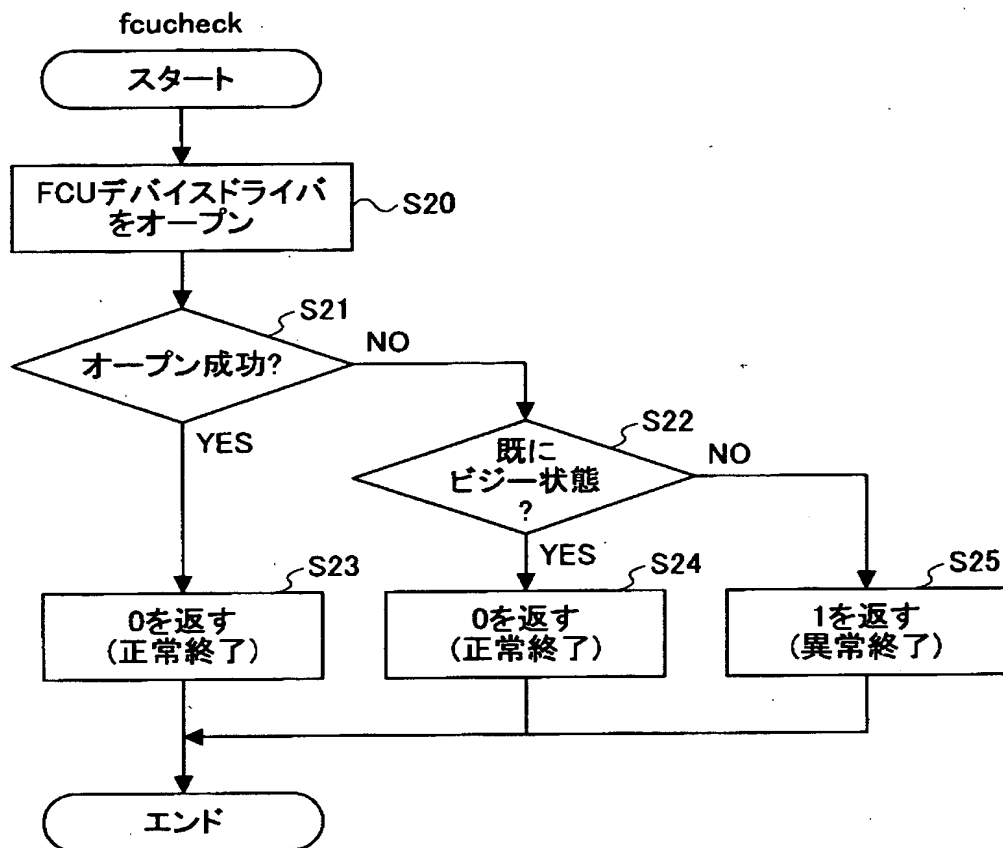
```
exec -c cpucheck1 /printer/bin/setfont_bitmap
```

```
exec -c cpucheck2 /printer/bin/setfont_vector
```

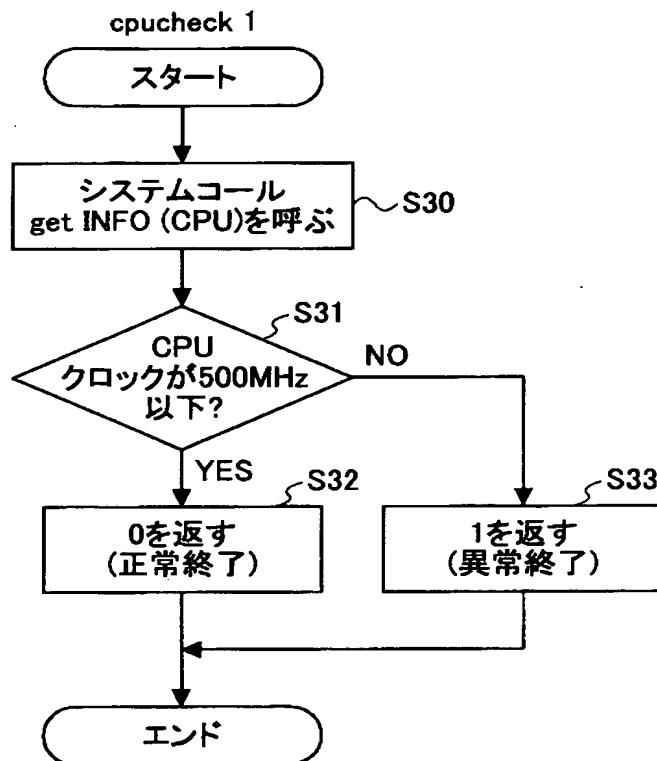
```
exec -c memcheck1 /web/bin/httpd -5
```

```
exec -c memcheck2 /web/bin/httpd -10
```

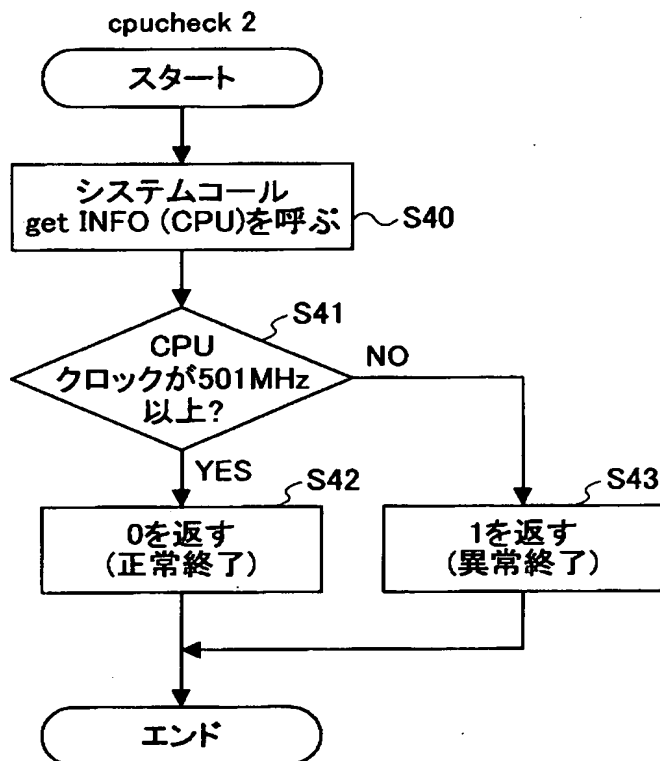
【図 7】

チェックプログラム「fcuccheck」
の処理の一例のフローチャート

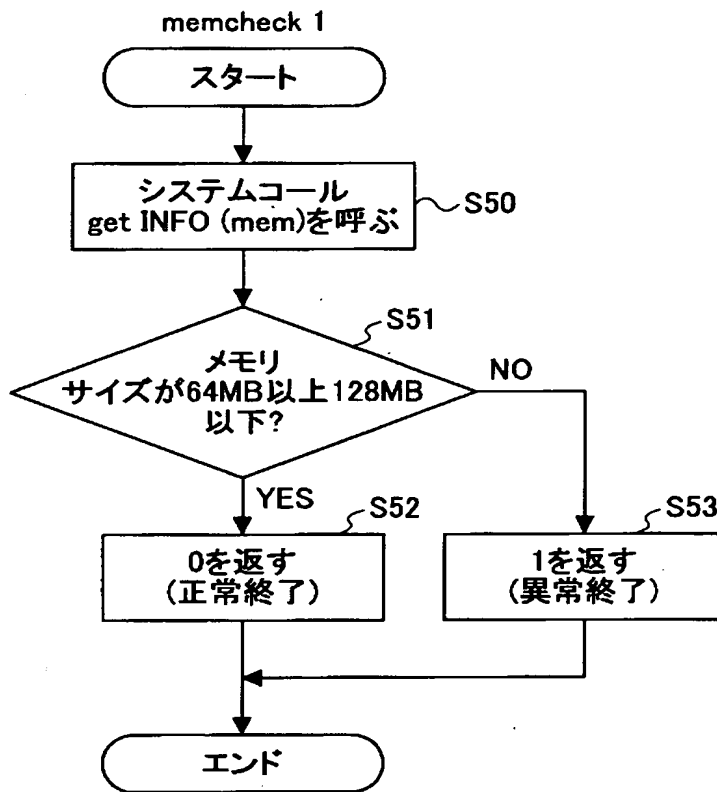
【図 8】

チェックプログラム「cpucheck1」
の処理の一例のフローチャート

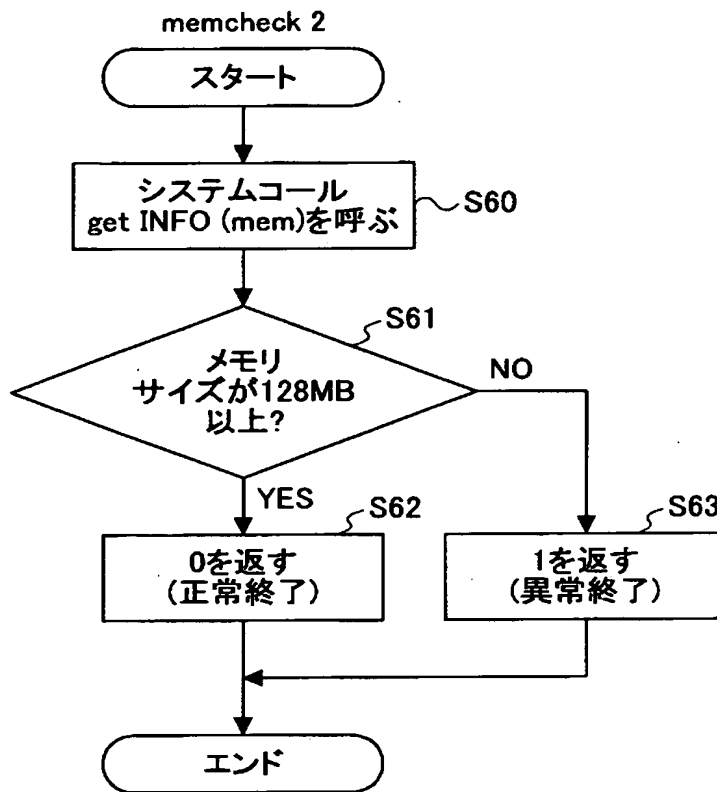
【図 9】

チェックプログラム「cpucheck 2」
の処理の一例のフローチャート

【図 10】

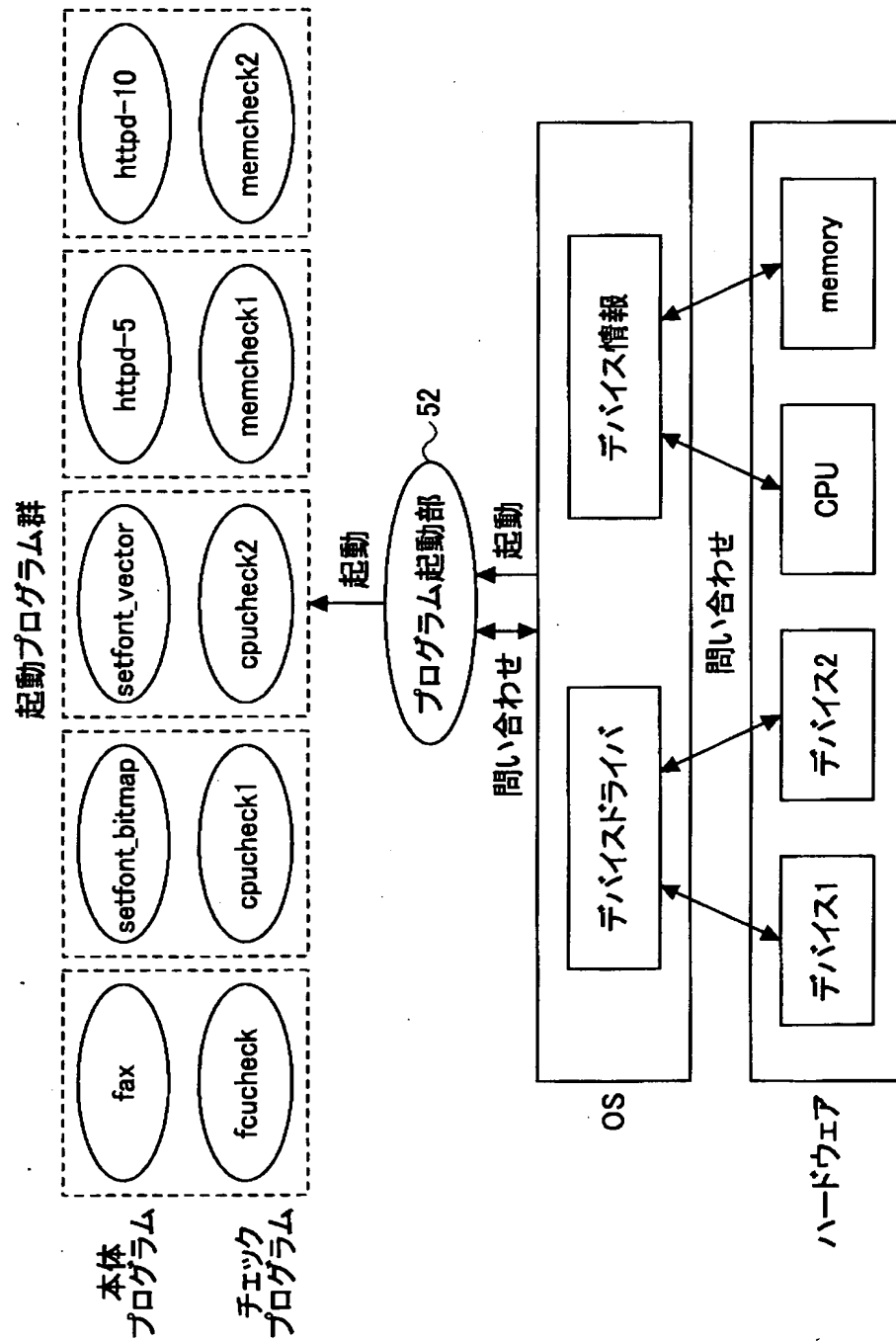
チェックプログラム「memcheck 1」
の処理の一例のフローチャート

【図 11】

チェックプログラム「memcheck 2」
の処理の一例のフローチャート

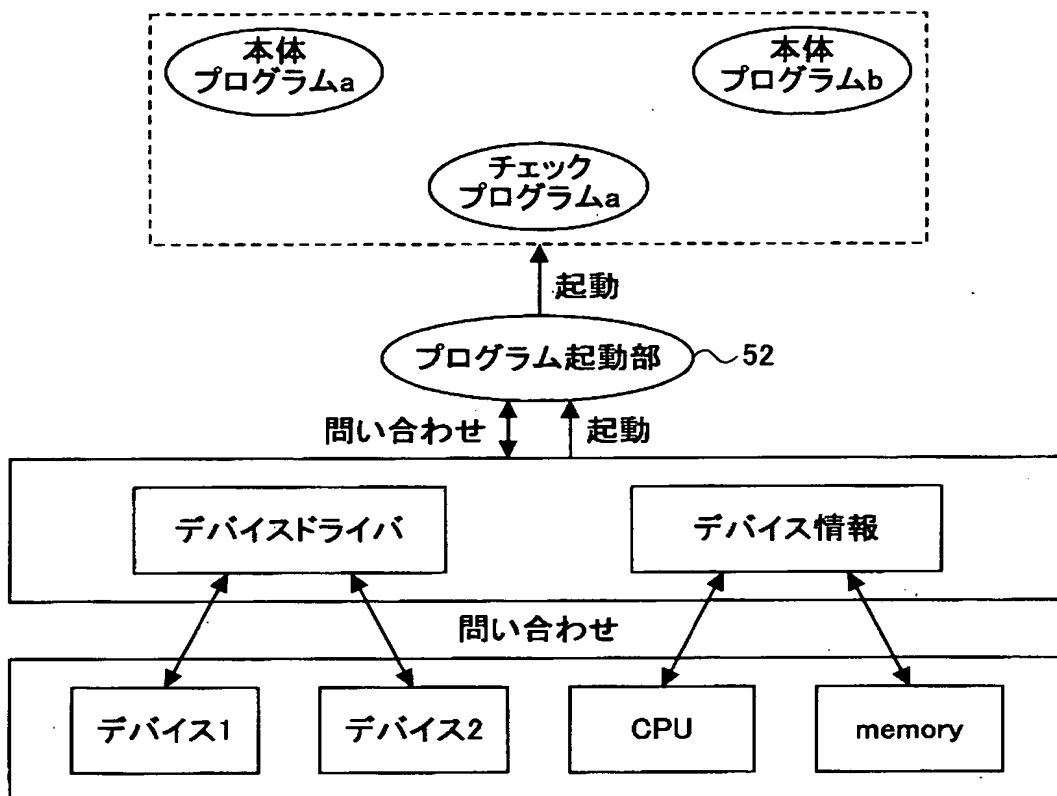
【図 12】

本体プログラムと、チェックプログラムと、プログラム起動部 52 と、OS と、ハードウェアとの関係を表した一例の関係図



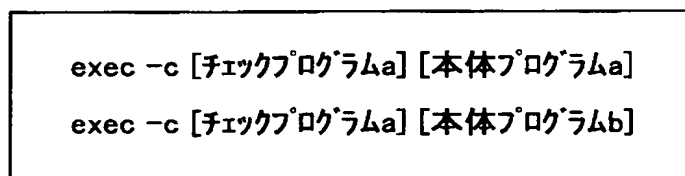
【図 13】

チェックプログラムと本体プログラムとが
1対nに対応する例について説明するための図



【図 14】

設定ファイルの他の一例の構成図



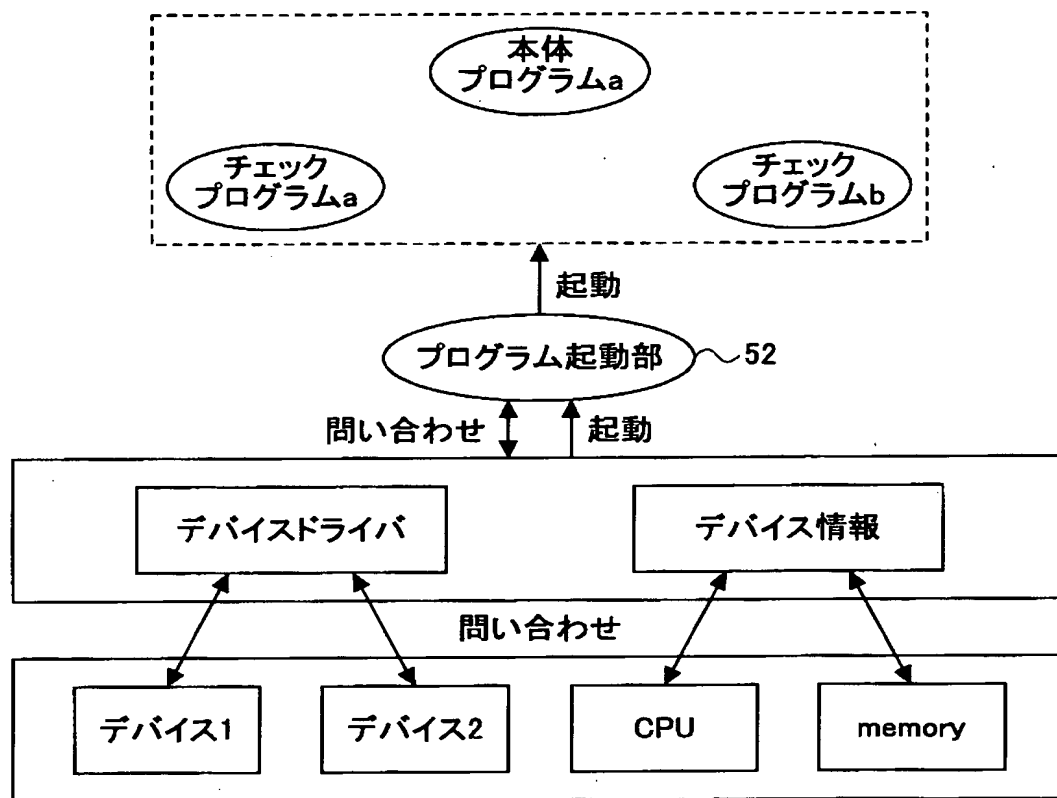
【図15】

マウントを抑制する設定ファイルの一例の構成図

```
mount -c memcheck3 romfs web.romfs /web
```

【図16】

チェックプログラムと本体プログラムとが
n対1に対応する例について説明するための図



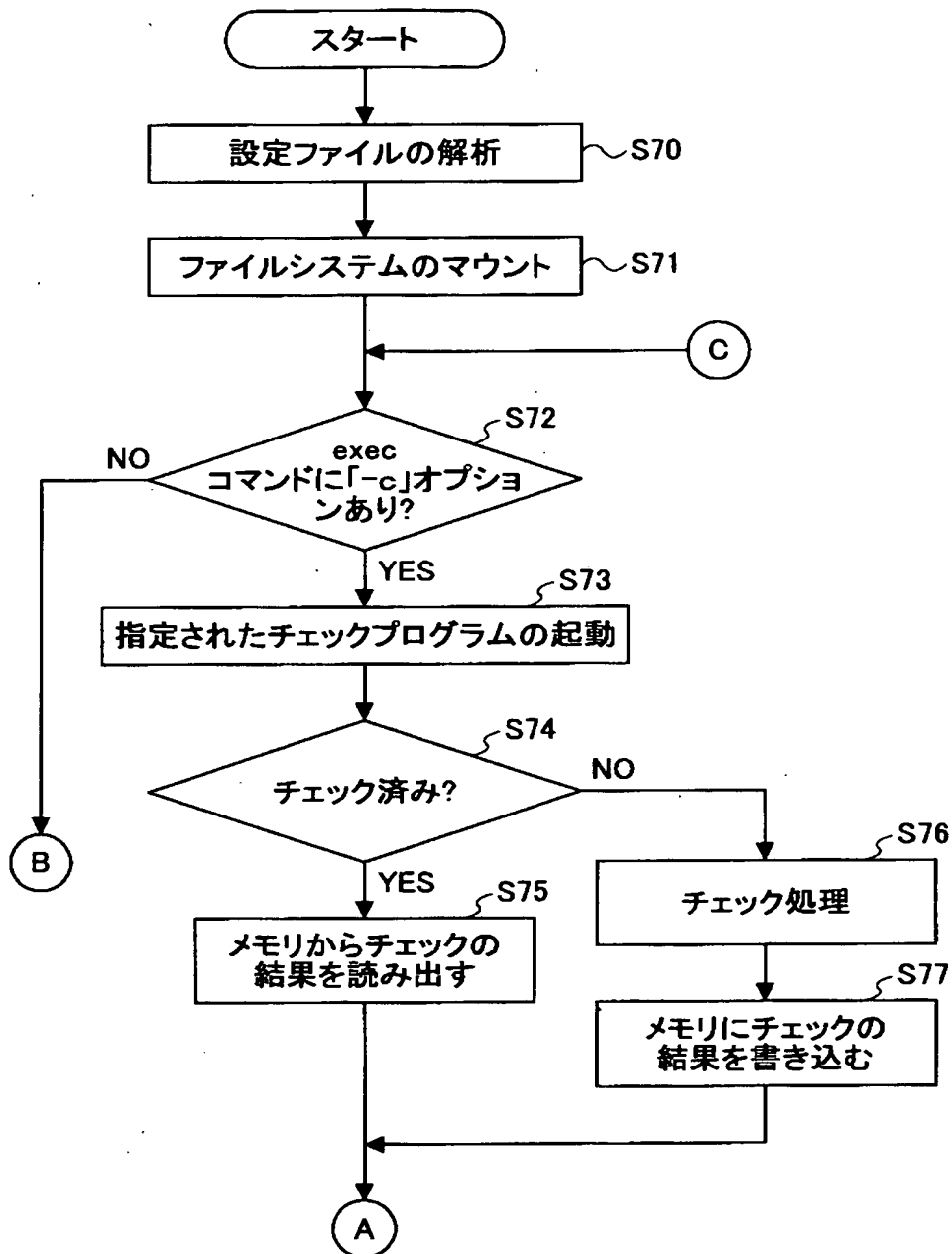
【図 1 7】

設定ファイルの他の一例の構成図

exec -c [チェックプログラムa]-c [チェックプログラムb] [本体プログラムa]

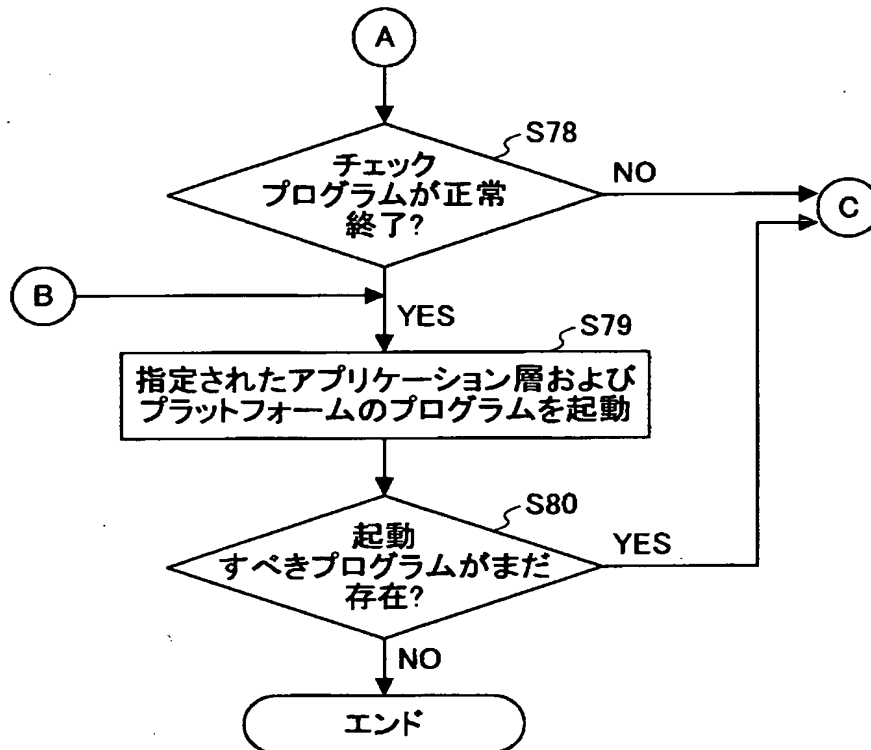
【図 18】

プログラム起動部およびチェックプログラムが行う
処理の一例のフローチャートである (1/2)



【図 19】

プログラム起動部およびチェックプログラムが行う
処理の一例のフローチャートである (2/2)



【書類名】 要約書

【要約】

【課題】 各プログラムの重複部分を削減することができ、ハードウェア資源に関連するプログラムを効率良く起動することが可能な画像形成装置およびプログラム起動方法を提供することを目的とする。

【解決手段】 画像形成処理で使用されるハードウェア資源と、画像形成に係る処理を行うプログラムとを有する画像形成装置であって、ハードウェア資源に関するチェックを行うチェック手段と、チェック手段とプログラムとの関連を設定する設定手段と、チェック手段によるチェックの結果に応じてチェック手段に関連するプログラムを起動する起動手段 52 とを有することにより上記課題を解決する。

【選択図】 図 1 2

特願 2002-342826

出 願 人 履 歴 情 報

識別番号

[000006747]

1. 変更年月日 1990年 8月24日
 [変更理由] 新規登録
 住 所 東京都大田区中馬込1丁目3番6号
 氏 名 株式会社リコー

2. 変更年月日 2002年 5月17日
 [変更理由] 住所変更
 住 所 東京都大田区中馬込1丁目3番6号
 氏 名 株式会社リコー